Announcements

- The times and locations of office hours are posted on the “Office and Consulting Hours” page of the course website. Please email us at cs135@uwaterloo.ca to set up an appointment outside of these hours.
- Assignment 2 is due on Tuesday, September 24, at 9:00 pm.
- If you have not already done so, make sure to complete Assignment 0 before submitting any assignments!
- Drop deadline without penalty is on Tuesday, September 24
- Drop-Down to CS 115 deadline is on Wednesday, October 23

Goals of this tutorial

You should be able to...

- Understand the basics of Boolean Algebra.
- Implement Conditional Statements in Racket.
- Utilize String-Based functions.
- Understand Symbols.
Review: Boolean Valued Functions
To determine if the proposition "a < b" is true or false, we can write it as the Racket expression (< a b).

There are also functions for >, =, >=, <=.

We can also combine multiple boolean functions using special forms and, or, not.

Example: “4 < x < 20” = (and (< 4 x) (< x 20))

Review: Truth Table
What are the missing values?

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th>(or b c)</th>
<th>(and a (or b c))</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
</tr>
<tr>
<td>T</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>T</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>T</td>
<td>F</td>
<td>T</td>
<td>F</td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>T</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>F</td>
<td>F</td>
<td>F</td>
<td></td>
</tr>
</tbody>
</table>

Review: String and Symbols
Symbolic data can be compared to other symbols
(symbol= ? 'Earth 'Mars) ⇒ false
(symbol= ? 'Earth 'Earth) ⇒ true
(symbol= ? 'Earth 'earth) ⇒ false (Why?)

String data can be compared in more flexible ways
(string= ? "YEET" "yeet") ⇒ false (Why?)
(string= ? "YEET" (string-upcase "yEeT")) ⇒ true
(string= ? "red" (substring "credit" 1 4)) ⇒ true
(= (string-length "") 0) ⇒ true
Clicker Question 1: True or False (or Error)

Assuming the following statements are defined, what will this expression produce?

\[
\text{(define (safe? s k)}
\begin{array}{l}
\text{(cond)}
\text{[and (> s 18) (< 18 k)] (cond)}
\text{[(<= (- k s) 5) true]}
\text{[else false])]
\text{[(and (< s 18) (> 18 k)) false]) (safe? 18.0 20)]}
\end{array}
\]

A True
B False
C Error

Problem 1: FizzBuzz

FizzBuzz is a classic Computer Science problem often used in interviews. Define a function that consumes a single number \( n \) and produces:

- “Fizz” if the number is a multiple of 3
- “Buzz” if the number is a multiple of 5
- “FizzBuzz” if it is a multiple of both
- \( n \) if it is not a multiple of either

The first 15 cases would look like
\( (1,2,\text{Fizz},4,\text{Buzz},\text{Fizz},7,8,\text{Fizz},\text{Buzz},11,\text{Fizz},13,14,\text{FizzBuzz}) \)

Problem 1: FizzBuzz - Design Recipe

- **Purpose:**
  \( (\text{fizzbuzz } n) \) Consumes a number \( n \) and produces the proper response following the rules of fizzbuzz.
- **Contract:**
  \( \text{fizzbuzz: Int } \rightarrow \text{ (Anyof Str Int) } \)
- **Example:**
  \( (\text{check-expect (fizzbuzz 3) } "\text{Fizz}" ) \)
- **Function Body...**
- **Tests:**
  \( (\text{check-expect (fizzbuzz } -5) \ "\text{Buzz}" ) \)
  \( (\text{check-expect (fizzbuzz 17) } 17) \)
  \( (\text{check-expect (fizzbuzz 30) } "\text{FizzBuzz}" ) \)
  \( (\text{check-expect (fizzbuzz 0) } "\text{FizzBuzz}" ) \)
Problem 2.1: Simplify Cond-Bool

;; (grt-bool location? cash?) Consumes a location and
;; cash and determines if you can take the bus home.
;; grt-bool: Bool Bool → Str

(define (grt-bool? location? cash?)
  (cond [location? "You're already home."
            [(cond [cash? false]
                  [else true])
             "You have to walk home."
             [else "Take the bus home."]])

Problem 2.1: Simplify Cond-Bool - Solution

;; (grt-bool-simple location? cash?) Consumes a location
;; and cash and determines if you can take the bus home.
;; grt-bool-simple: Bool Bool → Str

(define (grt-bool-simple? location? cash?)
  (cond [location? "You're already home."
            [not cash?] "You have to walk home."
            [else "Take the bus home."])

Problem 3: Echo

You have found yourself in a cave, a very large cave. When you shout you hear an echo of
yourself bouncing off the cave walls.

An echo is a string that repeats the same word twice (case insensitive). Define a function that
takes in a string and determines if it is an echo. (Will a string of odd length produce true?)

- (echo? "HELLOhello") ⇒ true
- (echo? "helloworld") ⇒ false
- (echo? "YeEtyEeT") ⇒ true
- (echo? "cs-cs") ⇒ false
- (echo? "") ⇒ true

Look for a string function that might help you with question.
(Help > Help Desk > Strings)
Problem 3 Bonus: Real Echo

Define a function where the first half of the string is uppercase, and second half of the string is lowercase.

(echo? "HELLOhello") ⇒ true
(echo? "helloworld") ⇒ false
(echo? "YeEtyEyT") ⇒ false
(echo? "cs-cs") ⇒ false
(echo? "") ⇒ true

Hint: Use Helper Functions